IOTIVITY AND EMBEDDED LINUX SUPPORT

Kishen Maloor
Intel Open Source Technology Center

OPEN INTERCONNECT CONSORTIUM℠

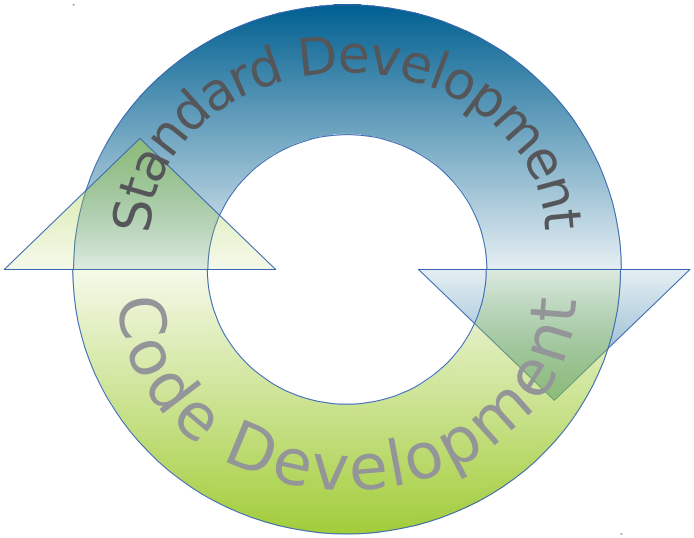
THE LINUX FOUNDATION

Outline

- Open Interconnect Consortium and IoTivity
- Software development challenges in embedded
- Yocto Project and how it addresses these challenges
- **Key takeaway:** IoTivity over Yocto makes an ideal platform for developing embedded IoT applications
- This is not a tutorial on Yocto

Open Interconnect Consortium

- Industry group with several member companies
- Interoperability standards for IoT devices
- IoTivity: Reference implementation



What is IoTivity?

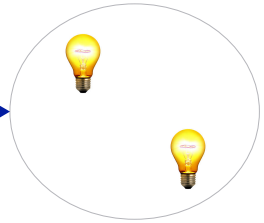
- Internet Of Things
  - Interconnecting physical objects with the digital world
  - Widespread deployment of Low Power Embedded computers

- IoTivity
  - High-level APIs for IoT Application Developers
  - Exposing “things” as resources
  - Discovering and manipulating resources over multiple network transports
  - Utilize emerging IoT technologies

Simple Use Cases

Smartphone

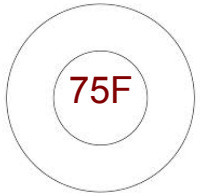
Turn Lights ON



Light bulbs with BLE radios

Notify Current Setting

Smart TV

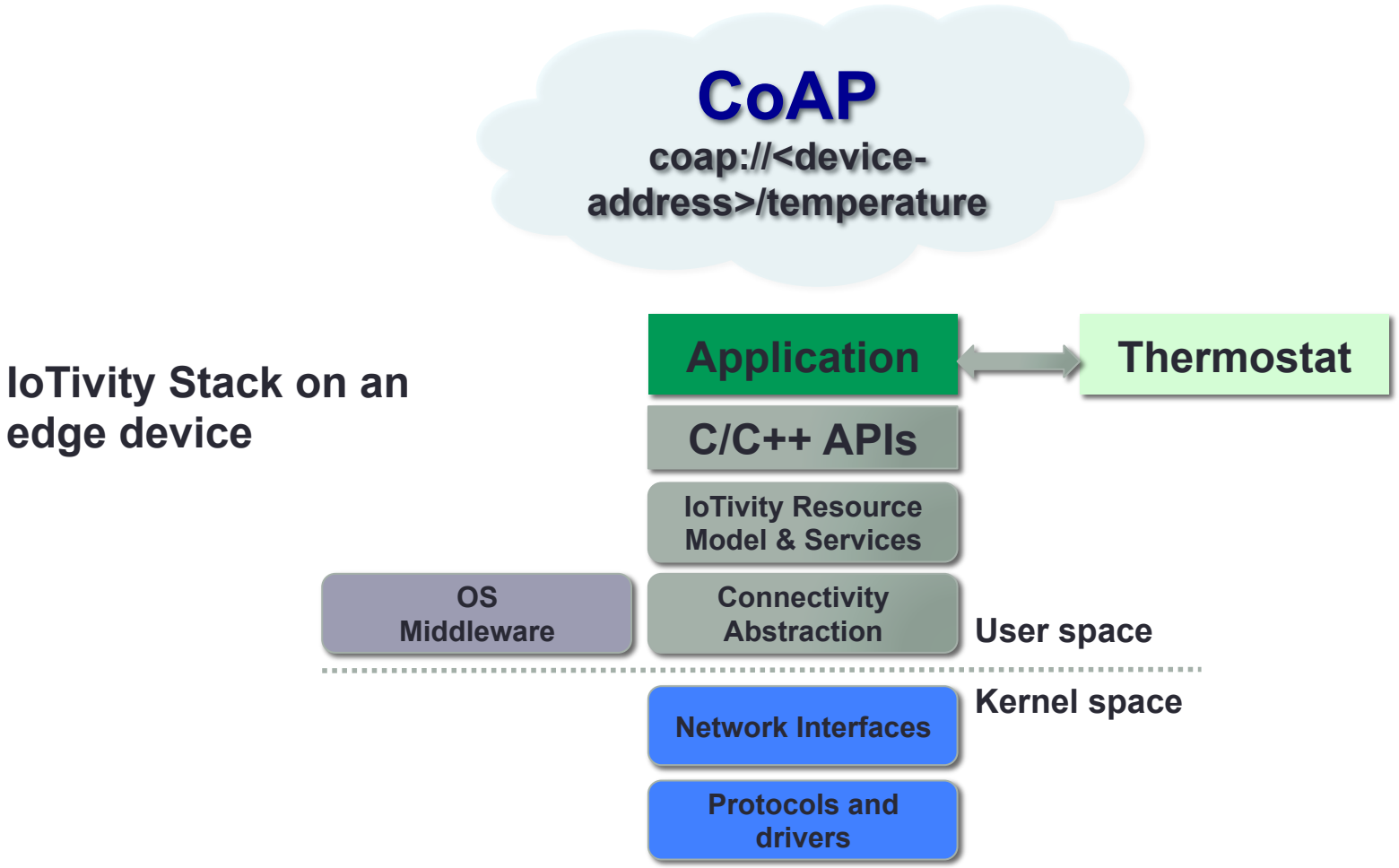


Digital Thermostat

Tablet

Regulate Temperature

IoTivity Software Stack



Emerging Open IoT Protocols

- 6LoWPAN: IPv6 over Low Power Wireless Personal Area Networks
- Bluetooth Smart
- IPSP
- RPL: Routing over Low Power and Lossy Networks
- …

- New RFCs being published followed by prototype Linux implementations
- Growing influence of Linux in IoT

Challenges

- Heterogeneous nature of targets, CPUs, kernels
  - **IoTivity needs to be ported to each and maintained separately. Not easily scalable.**
- IoT rapidly evolving with new protocols
  - **Need modular approach to quickly plug-in new IoT protocol implementations**

Challenges

- Embedded development now becoming mainstream with IoT
  - **Need cohesive software development infrastructure that is uniform across multiple IoT targets**

- **These challenges are addressed by the Yocto Project…**

Yocto Project

- http://www.yoctoproject.org/
  - Hosted at the Linux Foundation
  - Create customized OS images for embedded targets
  - Ready-to-use BSPs for multiple platforms
- Layer-based flexible build architecture
- Focus on configurability and reuse
- Support for major CPU architectures

Yocto Recipes

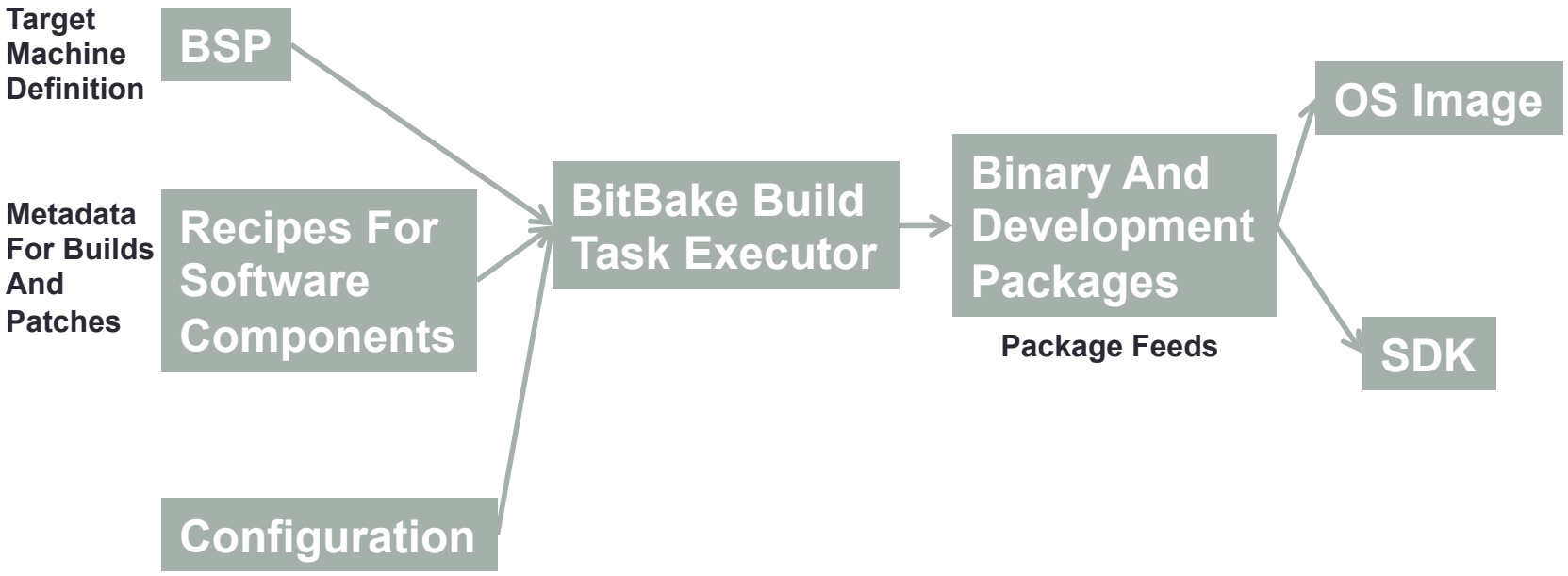
- Spec files with .bb extension
- Represents a “meta package”
- Define contents of binary and development packages
- Dependency relationships between recipes
- Versioning
- Interfaces for fetch, patch, configure, compile, install steps
- Architecture specific switches

Software Layers

- Related collections of recipes to build applications and middleware
- Customize build and configuration of BSP and other software layers
  - Recipes with .bbappend extension

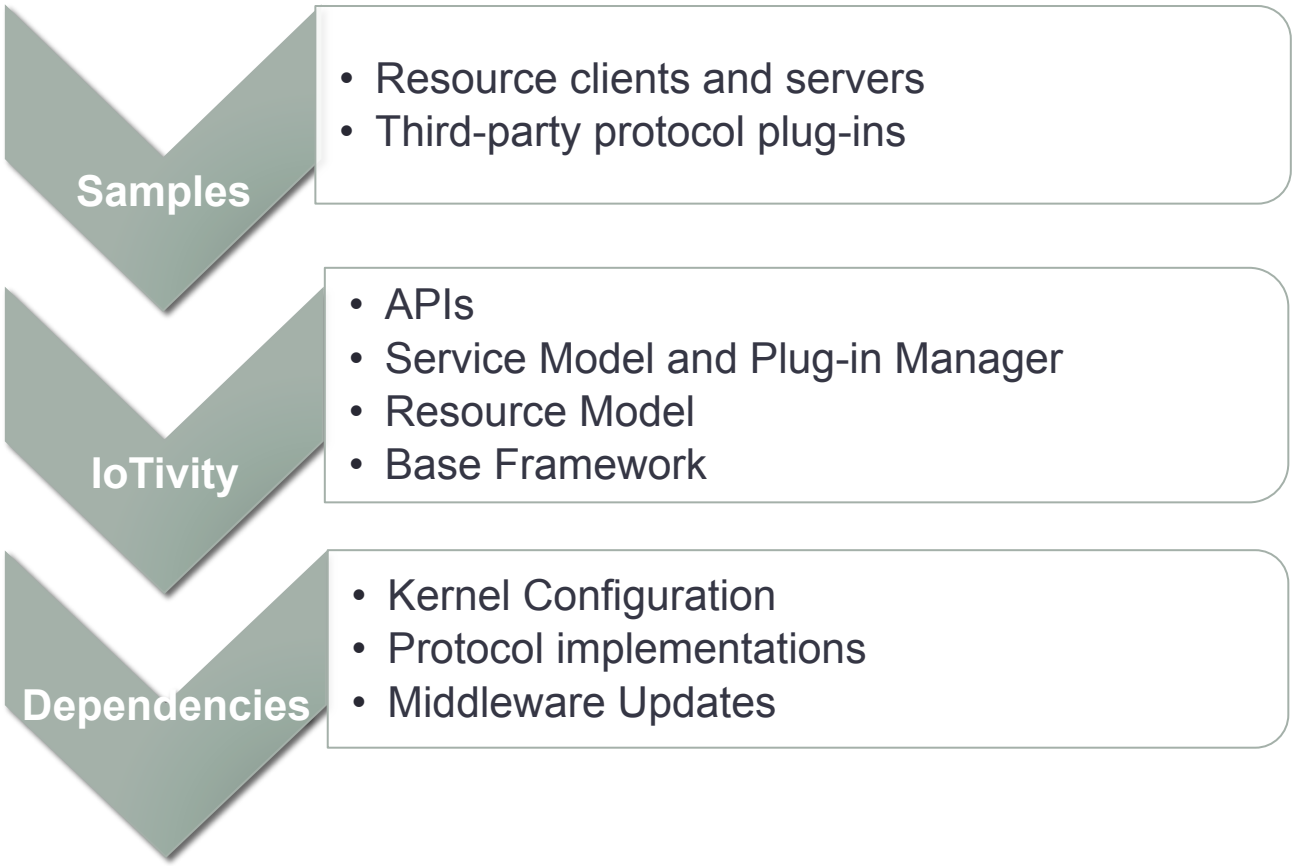
- **Package up IoTivity and dependencies in a target agnostic way**

Yocto Build Workflow



meta-oic Software Layer

- git://git.yoctoproject.org/meta-oic



Kernel Builds In Yocto

- linux-yocto
  - Upstream kernel based trees maintained by the Yocto Project
  - Platform specific branches
  - Recipes for respective kernels
- linux-yocto-custom
  - Build any git-based kernel

Adding Kernel Features

- Create a linux-yocto.bbappend recipe to customize the kernel
  - Resides in your layer and distributable
- Patches
- Configuration Fragments
  - Create a .cfg and place in your kernel .bbappend

```
#Enable features for IoTivity
CONFIG_BT_6LOWPAN=y
CONFIG_IEEE802154=y
CONFIG_IEEE802154_6LOWPAN=y
CONFIG_6LOWPAN_IPHC=y
CONFIG_MAC802154=y
```

Other Supporting Features

- Distribute new features as patches
  - Middleware
    - Adding a GATT interface for IoTivity to BlueZ
      - Create a .bbappend for the BlueZ recipe
  - Protocol integration
    - RPL (Routing protocol for Low Power and Lossy Networks)
    - XBee module for 802.15.4 support
  - Security related features

- **Opportunity to pack in early implementations of IETF specs via patches**

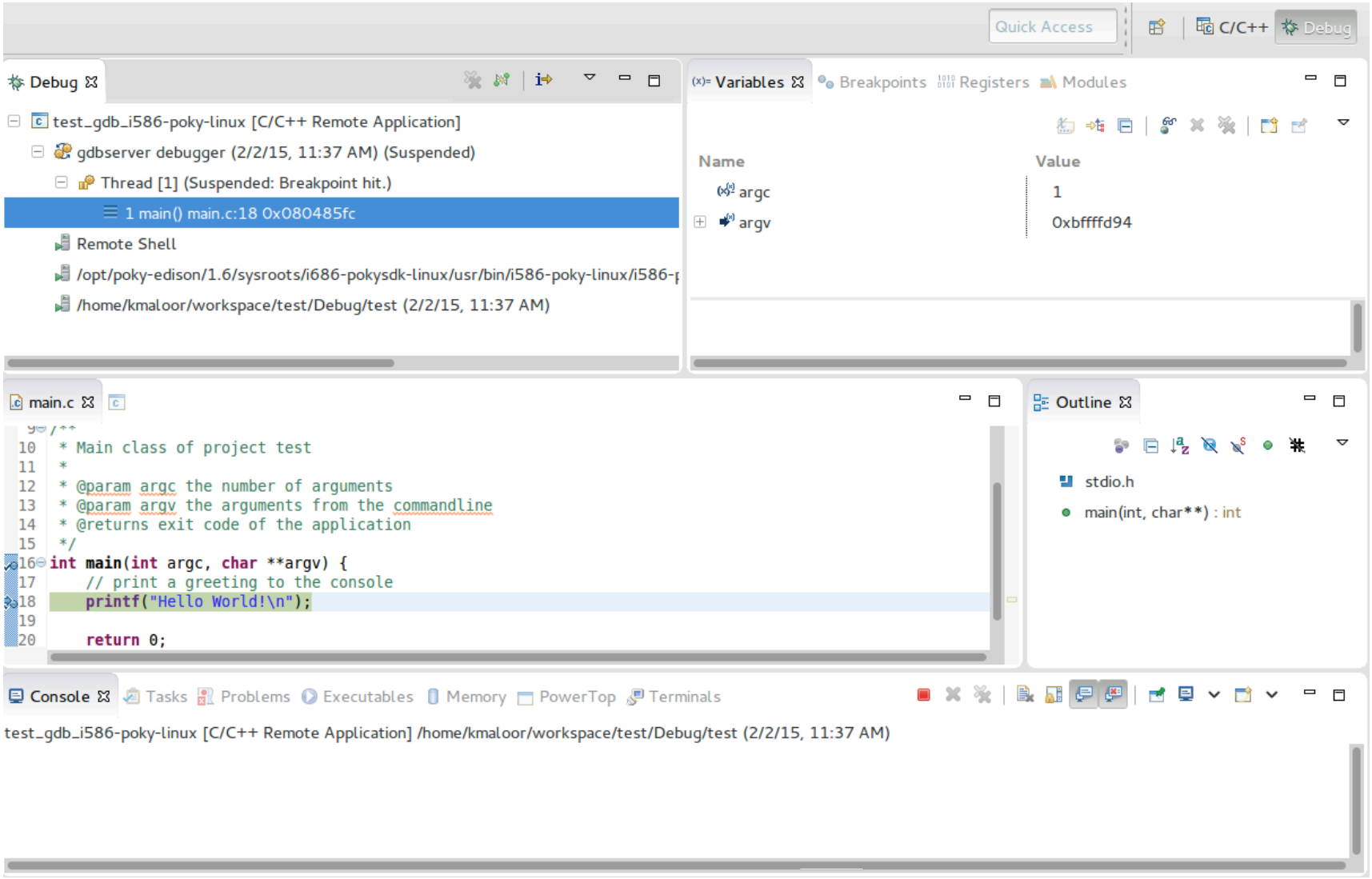
Application Development

- Application Development Toolkit
  - Standalone cross-compiling toolchain with debugging and profiling tools
- Constructing an SDK
  - Picks all development packages for target
    - ADT will include IoTivity SDK
  - Generates target ADT for specified build machine architecture
- **IoTivity developers can focus on application development without getting bogged down by details of target**

Yocto Eclipse Plug-in

- Eclipse integration with Yocto ADT
  - Access to cross-compiler, debugging and profiling tools
- Remote application debugging, step through code
  - Real hardware via network using its IP address
  - QEMU
- Install Plug-in and point it to your target ADT
  - Configure remote connection in “C++ Remote Application” under “Debug Configurations”

Remote Debugging

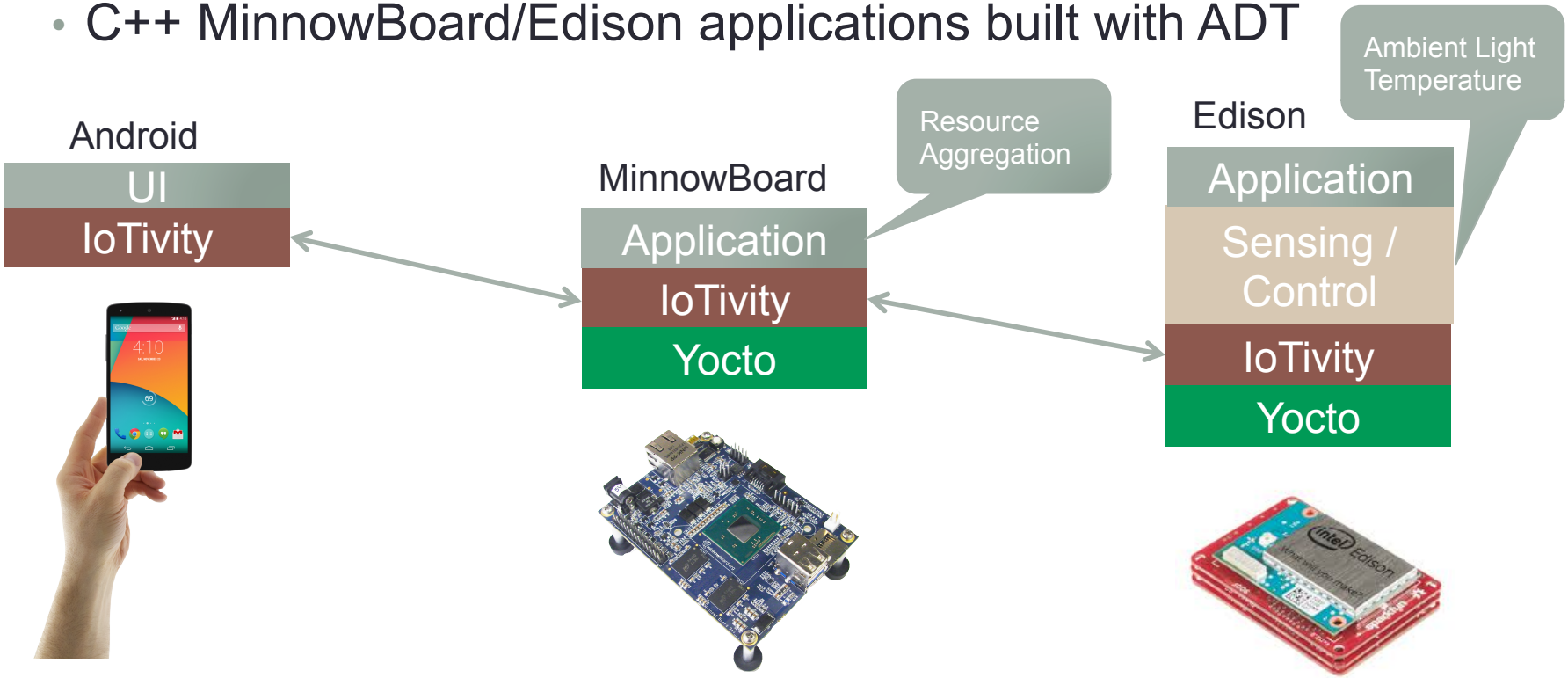


Releasing Your Application

- Write a recipe to build your application in the Yocto environment
- Distribute application packages for specific target platforms

Putting It To Test

- Built IoTivity and toolchains for Intel Edison and MinnowBoard MAX
  - BSPs available online
  - C++ MinnowBoard/Edison applications built with ADT

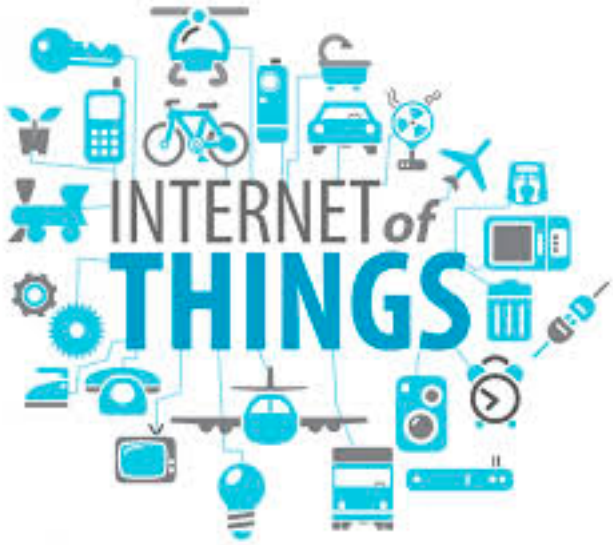


To Conclude…

- Yocto provides for greater scale
  - Configure in one place, deploy on any Yocto-based platform
- Improved embedded IoT app developer experience
- Linux supports state-of-the-art IoT technologies
- We've had promising results

IoTivity

yocto PROJECT



How Can You Participate In IoTivity?

- IoT application developers
- Open-source contributors
- Propose new framework features, use cases
- https://www.iotivity.org/get-involved
- IoTivity Mailing List

Resources

- IoTivity SDK and Samples https://www.iotivity.org/
- Open Interconnect Consortium http://openinterconnect.org
- meta-oic Yocto Layer https://git.yoctoproject.org/cgit/cgit.cgi/meta-oic/about/
- Working with kernels in the Yocto Project: Presentation https://www.yoctoproject.org/sites/default/files/devday-kernel-tzanussi-elc-2013.pdf
- Yocto Eclipse IDE Plug-in: Instructional video http://www.youtube.com/watch?v=3ZlOu-gLsh0

Thanks for your time!

Q&A